REMARKS

With this paper, independent method claims 1, 10, 17 and 21, and corresponding computer program product claim 25 have been amended, together with various dependent claims (2, 7-8, 11-12, 18, 22, 25-26, 30, 31, 33), while claims 6, 19, and 29 have been cancelled and claims 35-38 have been added. Accordingly, claims 1-5, 7-18, 21-28, and 30-38 are presented for reconsideration, of which claims 1, 10, 17, 21, 25, and 37 are independent.

The most recent Office Action ("*Office Action*") objected to dependent claims 5 and 6, as well as claims 28 and 29, as being substantial duplicates of each other. As is reflected in the above claim listing, the objected-to claims 6 and 29 have each been cancelled, and, as such, the object of record is now moot.

In addition, the *Office Action* rejected claims 1-3, 7-13. 16-29, 21-26, and 30-34 under 35 U.S.C. § 103(a), as being unpatentable over U.S. Patent Publication No. 2002/0188610 ("*Spencer*") in view of U.S. Patent No. 6,889,229 ("*Wong*").[1] The *Office Action* also rejected various dependent claims under 35 U.S.C. § 103(a) as obvious over Wong in combination with U.S. Patent No. 6,578,068 ("*Bowman-Amuah*").

In general, *Wong* describes a method for "peer-to-peer replication of objects" between various nodes connected over a network. To this end, *Wong* discloses "mapping" various object attributes from one sending node to a receiving node, so that unique data fields (and corresponding data) at the sending node can be correlated to data fields at the receiving node. If properly mapped and correlated, the receiving node can then receive (*i.e.*, copy over) and process data from the sending node. Otherwise, if not properly mapped, the data from the receiving node may not be properly copied at all, and, at the very least, cannot be processed. *Compare* col. 3, ll. 10-15, *with* col. 4, l. 49 – col. 5, l. 4; col. 9, ll. 25-30.

According to *Wong*, therefore, the sending node creates a replication group of objects, including any user-defined objects. Col. 6, ll. 41-56; Col. 8, ll. 25-43. Thereafter, the sending node copies data defining the user-defined object(s) to a data structure on the second node. Col. 7, ll. 58-65; col. 9, ll. 25-31. This data generally includes naming correlation/mapping

[1] Since Wong qualifies as "prior" art, if at all, under 35 U.S.C. 102(e), applicants reserve the right to challenge the status of that reference as qualifying "prior" art. Accordingly, any statement or comment herein to Wong is made merely for purposes of argument, and assumes *arguendo* that the reference is proper qualifying prior art.

information, such as column designations for the user-defined type, names for the user-defined type, attribute correlations between the sending and receiving node(s), and so forth. Col. 8, ll. 55-65; col. 15, ll. 3-36. Only after sending this naming correlation (or "mapping") information to the receiving node can the sending node copy the user-defined object to the receiving node without failure, since the receiving node would then know where to place and/or otherwise organize the received data (*e.g.*, the receiving node can correlate a column name in which to place the user-defined data.) (Col. 7, ll. 58-65; col. 9, ll. 25-31; col. 9, ll. 46-51; col. 11, ll. 29-41; col. 19, ll. 51-65)

Importantly, however, the *Wong* reference discloses nothing about the receiving node's ability to *process* data associated with a particular data type, such as data specifically-formatted to be opened only by a particular type of application program. In particular, the sole reason the receiving node in the *Wong* reference appears unable to process certain data appears to be that the receiving node does not know where to place (*i.e.*, "fill" data fields with received data) certain unique (user-defined) data it receives from the sending node. Col. 3, ll. 10-15; col. 7, ll. 48-51; col. 9, ll. 45-51.

This contrasts in at least one respect with Applicant's claimed methods, which require a middle tier server to receive an "extended assembly" from a back end server. For example, upon encountering an unfamiliar data type, a middle tier server can receive program code means or computer-executable code, such as an application program or module, which allows the middle tier server to recognize the data and then process it. At the outset, therefore, and in contrast with the *Wong* reference, the middle tier server can actually receive any data successfully from another source at any time, even if the receiving middle tier server does not recognize the data.

As such, the *Wong* reference fails to disclose, teach, or otherwise suggest, whether singly or in combination with *Spencer* or *Bowman-Amuah*, an extended assembly that includes "data obtained from the special table, including data identifying the data type, one or more definitions of the data type, and the code for enabling *processing* of data corresponding to the data type," as recited in amended claim 1. In addition, the *Wong* reference fails to disclose, teach, or otherwise suggest, whether singly or in combination with *Spencer* or *Bowman-Amuah*, that the extended assembly includes "*executable code* that, when executed, enables the one or more middle tier servers to *process* the modified stored data or the new stored data associated with the data type,"

as recited in amended claim 10. Similar amendments are found in independent claims 17, 21, and 25.

As previously mentioned, the *Wong* reference teaches that data having user-defined types cannot even be copied to a receiving node unless the user-defined types (and/or corresponding attribute data) are correlated or mapped in some way with data fields at the receiving node. Thus, the *Wong* reference necessarily fails to disclose, teach, or otherwise suggest, whether singly or in combination with *Spencer* or *Bowman-Amuah* (and even teaches away from), determinations at the back end server "based at least in part on a request by the new middle tier server for data to enable use of one or more data types," as recited in amended claim 17. Similar limitations are found in amended dependent claims 2, 7, 12, 18, 22, 26, and 30.

Along these lines, Applicants have added new claims 35-38, which include new independent claim 37, to clarify Applicants' invention. Claims 37-38 are drafted particularly from the perspective of what steps are performed at or by a given middle tier server that encounters a data type it cannot recognize.

As a final matter, the *Office Action* cited column 14, ll. 23-34 for the limitation recited in claim 34, which generally recites that the extended assembly is a single data structure that includes all of the data required to use a particular type. Applicants respectfully traverse this characterization of the *Wong* reference, particularly as an "extended assembly" should be properly construed. For example, Applicant's disclosure teaches that an extended assembly is an application program or module that is used by a middle tier server to "process" data associated with a data "type," such as "types" of image data formats, word processing data formats, etc. *Compare* ¶¶ 23-24, *with* ¶ 43. In general, the term "assembly" is understood in the art as computer code that can be (or is) assembled or compiled into computer-understandable information for directing specified computer processes. Correlated to Applicant's invention, this means that if a middle tier server encounters digital image data for which it does not have the particular assembly (*e.g.*, application program or module) to process/open it, regardless of whether the digital image data is placed or filled in an appropriate database data structure, the middle tier server cannot process the digital image data.

By contrast, a "type," as understood from the *Wong* reference, appears to mean a kind of end-user data category that is meant to be interpreted or modified via end-user input/output, rather than, for example, as computer-readable data that is used by a computer-system to process

data objects. For example, "Table 1" of the *Wong* reference shows that a "user-defined Type" includes the category of "street_address." As previously discussed, therefore, the information received from the sending computer simply maps various end-user-defined *categories*, and thus does not constitute a program or module used to direct a computer's processing of specific types/formats of computer-readable data objects (*e.g.*, new claim 38).

Even assuming, *arguendo*, that the extended assembly of Applicant's invention was analogous in some way to the information received from the sending node in the *Wong* reference, col. 14, ll. 23-34 does not teach a "single data structure" that can be used for all data associated with a data type. In particular, col. 14, ll. 23-34 discusses that a "single auxiliary leaf" (*i.e.*, for a "top-level" type) can be used as some sort of an index to indicate "the presence or absence" of "all contained user-defined types." Col. 14, ll. 25-27. As such, this single leaf does not contain the actual data (e.g., column number, column names, column field data) of other leafs, and can only be used in the context of other leafs received from the sending node. Applicants find no other teaching in the *Wong* reference to contradict this construction. Applicants, therefore, respectfully traverse the rejection of claim 34.

Thus, for at least the foregoing reasons, the rejection(s)/objection(s) of record are now moot.

Applicants therefore respectfully request favorable reconsideration and allowance of the pending claims. In the event the Examiner finds any remaining impediment to allowance that may be clarified through a telephone interview, the Examiner is requested to contact the undersigned attorney.

Dated this ____ day of _____, 2006.

Respectfully submitted,

MICHAEL J. FRODSHAM
Registration No. 48,699
Attorney for Applicant
Customer No. 047973

MJF
AAM0000000410V001